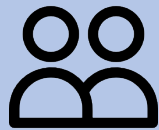


## Taller en Sala Nro. 11 Implementación de Grafos



En la vida real, los grafos se utilizan en videojuegos para representar las rutas que pueden tomar los caracteres, como por ejemplo League of Legends y World of Warcraft, así <https://goo.gl/images/R8WyGP>



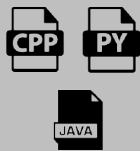
Trabajo en  
Parejas



Hoy, plazo  
máximo de  
entrega



Docente entrega  
código suelto en  
GitHub



Sí .cpp, .py  
o .java



No .zip, .txt,  
html o .doc



Alumnos  
entregan  
código suelto  
por GitHub

## Ejercicios a resolver

1. Usen la clase abstracta *Digraph*, llámenla *DigraphAM* e implementen grafos con la estructura de datos Matrices de Adyacencia Etiquetadas
2. Luego, creen la clase *DigraphAL* e implementen grafos con la estructura de datos Listas de Adyacencia. Ambas clases heredan la clase abstracta *Digraph*.
3. **[Ejercicio Opcional]** Generen una representación usando Graphviz de los grafos anteriores.

# Ayudas para resolver los Ejercicios

**Ayudas para el Ejercicio 1.....**

**[Pág. 3](#)**

**Ayudas para el Ejercicio 2.....**

**[Pág. 4](#)**

**Ayudas para el Ejercicio 3.....**

**[Pág. 5](#)**

## Ayudas para el Ejercicio 1



**Pista:** Vean en Guía de Laboratorios, numeral 4.10, “*Cómo hacer clases abstractas*”



**Error Común 1:** El método `getSuccessors` debe retornar los sucesores los identificadores de los vértices, no los pesos de los arcos.

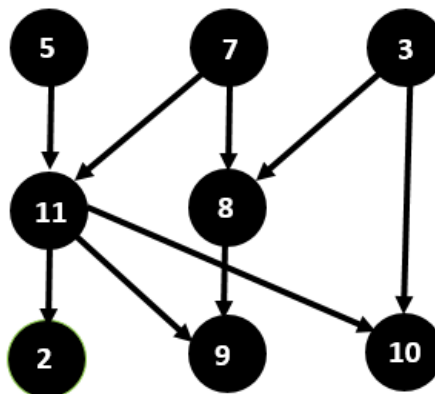


**Error Común 2:** El método `getSucessors` retorna los vecinos de un nodo, es decir, los nodos adyacentes. No es un recorrido en profundidad.



**Como un ejemplo,** para el grafo de la imagen, esta debe ser la matriz:

Grafo



**Matriz**

	2	3	5	7	8	9	10	11
2	0	0	0	0	0	0	0	0
3	0	0	0	0	1	0	1	0
5	0	0	0	0	0	0	0	1
7	0	0	0	0	1	0	0	1
8	0	0	0	0	0	1	0	0
9	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
11	1	0	0	0	0	1	1	0

## Ayudas para el Ejercicio 2



**Pista:** Vean en Guía de Laboratorios, numeral 4.8, “Cómo definir una clase pareja en Java”




**Error Común 1:** El método *getSuccessors* debe retornar los identificadores de los vértices, no los pesos de los arcos.



**Error Común 2:** El método *getSucessors* retorna los vecinos de un nodo, es decir, los nodos adyacentes. No es un recorrido en profundidad.



**Como un ejemplo,** para el grafo de la imagen del punto 1, esta debe ser la representación con listas:

	UNIVERSIDAD EAFIT ESCUELA DE INGENIERÍA DEPARTAMENTO DE INFORMÁTICA Y SISTEMAS	Cód. ST0245
		Estructuras de Datos 1

2 →	8 → 9
3 → 8,10	9 →
5 → 11	10 →
7 → 8, 11	11 → 2, 9, 11

La implementación de los puntos 3-6 se hacen en la clase "Recorridos.java".

## Ayudas para el Ejercicio 3



**Pista:** Para visualizar el grafo, utilice <http://www.webgraphviz.com/>



**Como un ejemplo** para grafo no dirigido:

```
graph {
  a -- b;
  a -- c;
  a -- e;
  b -- c;
  c -- d;
  c -- e;
}
```



**Como un ejemplo,** para grafo dirigido:

```
digraph G {
  "Welcome" -> "To" [label = "a"]
  "To" -> "Web"
  "To" -> "GraphViz!"
}
```

DOCENTE MAURICIO TORO BERMÚDEZ  
Teléfono: (+57) (4) 261 95 00 Ext. 9473. Oficina: 19 - 627  
Correo: [mtorobe@eafit.edu.co](mailto:mtorobe@eafit.edu.co)



**Error Común:**



# ¿Alguna inquietud?

## CONTACTO

**Docente Mauricio Toro Bermúdez**

**Teléfono:** (+57) (4) 261 95 00 **Ext.** 9473

**Correo:** mtorobe@eafit.edu.co

**Oficina:** 19- 627

Agende una cita con él a través de **<http://bit.ly/2gzVg10>** , en la pestaña *Semana*. Si no da clic en esta pestaña, parecerá que toda la agenda estará ocupada.